RECEIVED
CENTRAL FAX CENTER

DEC 06 2006

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Appellants:	Karen L. NOEL et al.	§	Confirmation No.:	7142
Serial No.:	10/619,697	§	Group Art Unit:	2188
Filed:	07/15/2003	§	Examiner:	K. M. Patel
For:	Method And System Of Writing Data In A Multiple Processor Computer System	§	Docket No.:	200312434-1

I hereby certify that this paper is being transmitted to the U.S. Patent and Trademark Office facsimile number (571) 273-8300.
Date of facsimile: 12/06/2006
Typed Name: Mark E. Scott
Signature:

RESPONSE TO NOTIFICATION OF NON-COMPLIANT APPEAL BRIEF (37 CFR 41.37)

Mail Stop Appeal Brief – Patents
Commissioner for Patents
PO Box 1450
Alexandria, VA 22313-1450

December 6, 2006

Sir:

In response to the Notification of Non-Compliant Appeal Brief dated November 24, 2006, Appellants submit a revised Appeal Brief. In said Notice, the Examiner objected to Appellants' revised Appeal Brief filed September 25, 2006, because it allegedly "does not contain a concise statement of each ground of rejection presented for review (37 CFR 41.37(c)(1)(vi)." A revised Appeal Brief is submitted herewith that comports with this rule.

Respectfully submitted,

Mark E. Scott
PTO Reg. No. 43,100
CONLEY ROSE, P.C.
(512) 391-1900 (Phone)
(512) 320-9181 (Fax)
ATTORNEY FOR APPELLANTS

HEWLETT-PACKARD COMPANY
Intellectual Property Administration
Legal Dept., M/S 35
P.O. Box 272400
Fort Collins, CO 80527-2400

RECEIVED
CENTRAL FAX CENTER

DEC 06 2006

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Appellants:	Karen L. NOEL et al.	§	Confirmation No.:	7142
Serial No.:	10/619,697	§	Group Art Unit:	2188
Filed:	07/15/2003	§	Examiner:	K. M. Patel
For:	Method And System Of Writing Data In A Multiple Processor Computer System	§	Docket No.:	200312434-1

APPEAL BRIEF

Mail Stop Appeal Brief – Patents
Commissioner for Patents
PO Box 1450
Alexandria, VA 22313-1450

Date: August 31, 2006

Sir:

Appellants hereby submit this Appeal Brief in connection with the above-identified application. A Notice of Appeal was filed via facsimile on August 4, 2006.

TABLE OF CONTENTS

I. REAL PARTY IN INTEREST

The real party in interest is the Hewlett-Packard Development Company (HPDC), a Texas Limited Partnership, having its principal place of business in Houston, Texas. HPDC is a wholly owned affiliate of Hewlett-Packard Company (HPC). The Assignment from the inventors to HPDC was recorded on October 13, 2003, at Reel/Frame 014045/0256.

II. RELATED APPEALS AND INTERFERENCES

Appellants are unaware of any related appeals or interferences.

III. STATUS OF THE CLAIMS

Originally filed claims:	1-19.
Claim cancellations:	None.
Added claims:	None.
Presently pending claims:	1-19.
Presently appealed claims:	1-19.

RECEIVED
CENTRAL FAX CENTER

DEC 0 6 2006

IV. STATUS OF THE AMENDMENTS

No claims were amended after the final Office action dated May 4, 2006.

RECEIVED
CENTRAL FAX CENTER
DEC 06 2006

V. SUMMARY OF THE CLAIMED SUBJECT MATTER

The specification is directed to a method and system of writing data in a multiple processor computer system.[1] At least some of the illustrative embodiments are methods as in claim 1 comprising executing a first instance of a program on a first processor in a computer system having multiple processors[2] (and wherein the program refers to a virtual memory address (VMA) in a page table to obtain a pointer to a memory location to write writable data),[3] and executing a second instance of the program on a second processor in the computer system[4] (and wherein the second instance of the program refers to a VMA in a page table to obtain a pointer to a memory location to write the writable data).[5] The VMA referred to by each of the first and second instance of the program is the same, the VMA referred to by the first instance of the program points to a memory coupled to the first processor, and the VMA referred to by the second instance of the program points to a memory coupled to the second processor.[6]

Other illustrative embodiments are computer readable media, as in claim 8, the computer readable media storing programs executable by a processor that, when executed, perform a method comprising accessing a read/write variable in a computer system having a plurality of functional units[7] (each of the plurality of functional units having a processor and a random access memory (RAM) coupled to the processor),[8] the accessing by referring to a virtual

---

[1] Specification Title.

[2] Specification Paragraph [0014], lines 6-14. A shorthand notation for citations to the specification takes the for ([paragraph], [lines]). Thus, the illustrative citation of this foot note in the shorthand notation takes the form ([0014], lines 6-14).

[3] ([0016], lines 7-10), Figure 2, elements 46, 48, 50, 52.

[4] ([0014], lines 6-14).

[5] ([0016], lines 7-10), Figure 2, elements 46, 48, 50, 52.

[6] ([0016], lines 10-13), Figure 2.

[7] ([0022], lines 1-12).

[8] ([0013], lines 3-7), Figure 1, elements 38, 40, 42 and 44.

memory address (VMA) in a page table to locate the read/write variable[9] (wherein the VMA in each functional unit is the same, and wherein the VMA in each functional unit contains a pointer to RAM within its functional unit.)[10]

Other illustrative embodiments are computer systems as in claim 12 comprising a first processor coupled to a first memory (the first processor and first memory forming a first functional unit),[11] a second processor coupled to a second memory and forming a second functional unit (the second processor coupled to the first processor),[12] a page table in the first functional unit having a virtual memory address (VMA) for a read/write variable (the VMA in the page table of the first functional unit pointing to the first memory),[13] and a second page table in the second functional unit having a VMA for the read/write variable (the VMA in the page table of the second functional unit pointing to the second memory).[14]

Other illustrative embodiments are computer systems as in claim 16 comprising a first means for executing programs[15] coupled to a first means for storing programs and data[16] (the first means for executing and first means for storing forming a first functional unit),[17] a second means for executing programs[18] coupled to a second means for storing programs and data[19] (and forming a second functional unit,[20] the second means for executing coupled to the first

---

[9] ([0016], lines 7-10), Figure 2, elements 46, 48, 50, 52.

[10] ([0016], lines 10-13), Figure 2.

[11] ([0012], lines 1-4), Figure 1, elements 38, 40, 42 and 44.

[12] ([0012], lines 1-4), Figure 1, elements 38, 40, 42 and 44.

[13] ([0016], lines 7-13), Figure 2, elements 46, 48, 50, 52.

[14] ([0016], lines 7-13), Figure 2, elements 46, 48, 50, 52.

[15] This limitations is specifically identified as a means-plus-function limitation under 35 U.S.C. § 112, sixth paragraph; ([0010], lines 1-8) Figure 1, elements 16, 18, 22 and 24.

[16] This limitations is specifically identified as a means-plus-function limitation under 35 U.S.C. § 112, sixth paragraph; ([0010], lines 8-13), Figure 1, elements 12, 14, 26 and 28.

[17] ([0012], lines 1-4).

[18] This limitations is specifically identified as a means-plus-function limitation under 35 U.S.C. § 112, sixth paragraph; ([0010], lines 1-8). Figure 1, elements 16, 18, 22 and 24.

[19] This limitations is specifically identified as a means-plus-function limitation under 35 U.S.C. § 112, sixth paragraph; ([0010], lines 8-13), Figure 1, elements 12, 14, 26 and 28.

[20] ([0012], lines 1-4).

means for executing), a page table in the first functional unit having a virtual memory address (VMA) for a read/write variable (the VMA in the page table of the first functional unit pointing to the first means storing),[21] and a second page table in the second functional unit having a VMA for the read/write variable (the VMA in the page table of the second functional unit pointing to the second means for storing).[22]

[21] ([0016], lines 7-13), Figure 2, elements 46, 48, 50, 52.

[22] ([0016], lines 7-10), Figure 2, elements 46, 48, 50, 52.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Whether claims 1-2 and 8-19 are unpatentable under 35 U.S.C. § 103 by Harvey (U.S. Pat. No. 6,233,668).[23]

Whether claims 3-7 are unpatentable under 34 U.S.C. § 103 over Harvey in view of Backer (U.S. Pat. No. 6,266,745).

[23] The Office action of May 4, 2006 rejects claims 8-19 by stating, "Claim 8-19 are also rejected under the same rationale as applied to claims 1-7 above." Claims 1-2 are rejected as allegedly obvious over Harvey, and claims 3-7 are rejected as obvious over Harvey and Backer. Thus, it is not clear as to the nature of the rejections of claims 8-19, and so it is assumed that claims 8-19 are rejected as obvious over Harvey only.

RECEIVED
CENTRAL FAX CENTER
DEC 06 2006

VII. ARGUMENT

A. Section 103 Rejections over Harvey

1. Claims 1-2 and 8-19

Claims 1-2 and 8-19 stand rejected over Harvey. Claim 1 is representative of the claims of this grouping. The grouping should not be construed to mean the patentability of any of the claims may be determined in later actions (*e.g.*, actions before a court) based on the groupings. Rather, the presumption of 35 U.S.C. § 282 shall apply to each of these claims individually.

Harvey is directed to a system of concurrent page tables.[24] In particular, Harvey is directed to resumption of a particular process on a second processor (the second processor having local memory), where the particular process was previously executed on a first processor (the first processor having different local memory).[25] The difficulty addressed by Harvey is that page tables used when the particular process executed on the first processor may point to physical memory local to the first processor, but the same page tables used in the instantiation of the process in the second processor point to non-local memory of the second processor. In order to address this concern, Harvey discloses a system where "shared code and read-only data is replicated,"[26] and where there is a shared page table for the duplicated code and read-only data.[27] The precise mechanism used by Harvey to provided the shared page table is irrelevant to the rejections addressed in this Appeal Brief; however, of particular interest is that Harvey discusses the duplication of "shared code and **read-only** data," and that Harvey does not appear to address, or be operational with, duplication of writable data.

Illustrative claim 1, by contrast, specifically recites, "executing a first instance of a program on a first processor in a computer system having multiple processors, and wherein the program refers to a virtual memory address (VMA) in a page table to obtain a pointer to a memory location to write **writable data**;

---

[24] Harvey Title.

[25] Harvey Col. 8, lines 63-66.

[26] Harvey Col. 8, line 63 through Col. 9, line 1.

executing a second instance of the program on a second processor in the computer system, and wherein the second instance of the program refers to a virtual memory address (VMA) in a page table to obtain a pointer to a memory location to write **the writable data**; and wherein the VMA referred to by each of the first and second instance of the program is the same, and wherein the VMA referred to by the first instance of the program points to a memory coupled to the first processor, and wherein the VMA referred to by the second instance of the program points to a memory coupled to the second processor." Appellants respectfully submit that Harvey fails to teach or suggest such a system. In particular, Harvey only discusses duplicating "shared code and **read-only data.**" Thus, Harvey does not teach or suggest "executing a first instance of a program ... [that] refers to a virtual memory address (VMA) in a page table to obtain a pointer to a memory location to write **writable data**; executing a second instance of the program ... [that] refers to a virtual memory address (VMA) in a page table to obtain a pointer to a memory location to write **the writable data**; and wherein the VMA referred to by each of the first and second instance of the program is the same, and wherein the VMA referred to by the first instance of the program points to a memory coupled to the first processor, and wherein the VMA referred to by the second instance of the program points to a memory coupled to the second processor." In fact, Harvey appears to teach away from the claimed limitations.

Based on the foregoing, Appellants respectfully submit that the rejections of the claims in this grouping be reversed, and the claims set for issue.

B. Section 103 Rejections over Harvey in view of Backer

1. Claims 3-7

As for rejections over Harvey and Backer, Harvey does not expressly or inherently teach "executing a first instance of a program ... [that] refers to a virtual memory address (VMA) in a page table to obtain a pointer to a memory location to write **writable data**; executing a second instance of the program ... [that] refers to a virtual memory address (VMA) in a page table to obtain a pointer to a

---

[27] Harvey Col. 9, lines 10-13.

memory location to write **the writable data**; and wherein the VMA referred to by each of the first and second instance of the program is the same, and wherein the VMA referred to by the first instance of the program points to a memory coupled to the first processor, and wherein the VMA referred to by the second instance of the program points to a memory coupled to the second processor" as required by claim 1 (from which claims 3-7 depend). Thus, even if hypothetically the teachings of Backer are precisely as the Office action suggests (which Appellants do not admit), Harvey and Backer still fail to teach or suggest the limitations of claim 3-7 given the shortcoming of Harvey as discussed with respect to claims 1-2 and 8-19.

VIII. CONCLUSION

For the reasons stated above, Appellants respectfully submit that the Examiner erred in rejecting all pending claims. It is believed that no extensions of time or fees are required, beyond those that may otherwise be provided for in documents accompanying this paper. However, in the event that additional extensions of time are necessary to allow consideration of this paper, such extensions are hereby petitioned under 37 C.F.R. § 1.136(a), and any fees required (including fees for net addition of claims) are hereby authorized to be charged to Hewlett-Packard Development Company's Deposit Account No. 08-2025.

Respectfully submitted,

Mark E. Scott
PTO Reg. No. 43,100
CONLEY ROSE, P.C.
(713) 238-8000 (Phone)
(713) 238-8008 (Fax)
ATTORNEY FOR APPELLANTS

HEWLETT-PACKARD COMPANY
Intellectual Property Administration
Legal Dept., M/S 35
P.O. Box 272400
Fort Collins, CO 80527-2400

IX. CLAIMS APPENDIX

1. A method comprising:

executing a first instance of a program on a first processor in a computer system having multiple processors, and wherein the program refers to a virtual memory address (VMA) in a page table to obtain a pointer to a memory location to write writable data;

executing a second instance of the program on a second processor in the computer system, and wherein the second instance of the program refers to a virtual memory address (VMA) in a page table to obtain a pointer to a memory location to write the writable data; and

wherein the VMA referred to by each of the first and second instance of the program is the same, and wherein the VMA referred to by the first instance of the program points to a memory coupled to the first processor, and wherein the VMA referred to by the second instance of the program points to a memory coupled to the second processor.

2. The method as defined in claim 1 further comprising:

wherein the executing the first instance step further comprises executing the first instance of the program in a first functional unit of the multiple processor system;

wherein the executing the second instance step further comprises executing the second instance of the program in a second functional unit of the multiple processor system; and

wherein the first and second instances of the program are replicated versions of the same program.

3. The method as defined in claim 1 wherein the program is an operating system program, and wherein the writable data further comprises a performance counter count value.

4. The method as defined in claim 3 further comprising:

reading the count value from the memory coupled to the first processor;

reading the count value from the memory coupled to the second processor; and

combining the count values.

5. The method as defined in claim 4 wherein the performance counter count value is a number representing a number of page allocations in memory.

6. The method as defined in claim 4 wherein the performance counter count value is a number representing a number of disk accesses.

7. The method as defined in claim 1 wherein the program is an operating system program, and wherein the writable data further comprises a look-aside list header for process control blocks.

8. A computer readable media storing programs executable by a processor that, when executed, perform the following steps:

accessing a read/write variable in a computer system having a plurality of functional units, each of the plurality of functional units having a processor and a random access memory (RAM) coupled to the processor; the accessing by

referring to a virtual memory address (VMA) in a page table to locate the read/write variable, wherein the VMA in each functional unit is the same, and wherein the VMA in each functional unit contains a pointer to RAM within its functional unit.

9. The computer readable media as defined in claim 8 wherein the steps performed by the programs further comprise:

reading each of the read/write variables throughout the computer system;

combining the read/write variables; and

writing the combined read/write variables to a single location within the computer system.

10. The computer readable media as defined in claim 9 wherein the combining step further comprises adding the values of each of the read/write variables.

11. The computer readable media as defined in claim 9 wherein the steps performed by the programs further comprise clearing each of the read/write variables.

12. A computer system comprising:
a first processor coupled to a first memory, the first processor and first memory forming a first functional unit;
a second processor coupled to a second memory and forming a second functional unit, the second processor coupled to the first processor;
a page table in the first functional unit having a virtual memory address (VMA) for a read/write variable, the VMA in the page table of the first functional unit pointing to the first memory; and
a second page table in the second functional unit having a VMA for the read/write variable, the VMA in the page table of the second functional unit pointing to the second memory.

13. The computer system as defined in claim 12 further comprising:
a first replicated program executing on the first processor, the first replicated program writing the read/write variable at a location indicated by the VMA in the page table of the first functional unit;
a second replicated program executing on the second processor, the second replicated program writing the read/write variable at a location indicated by the VMA in the page table of the second functional unit; and

wherein the first and second replicated programs are the copies of a same program.

14. The computer system as defined in claim 13 wherein the first and second replicated programs are copies of an operating system program, and wherein the read/write variable is a counter that indicates a number of executions of a code path of the operating system program.

15. The computer system as defined in claim 13 wherein the first and second replicated programs are copies of an operating system program, and wherein the read/write variable is a look-aside list header for process control blocks.

16. A computer system comprising:

a first means for executing programs coupled to a first means for storing programs and data, the first means for executing and first means for storing forming a first functional unit;

a second means for executing programs coupled to a second means for storing programs and data, and forming a second functional unit, the second means for executing coupled to the first means for executing;

a page table in the first functional unit having a virtual memory address (VMA) for a read/write variable, the VMA in the page table of the first functional unit pointing to the first means storing; and

a second page table in the second functional unit having a VMA for the read/write variable, the VMA in the page table of the second functional unit pointing to the second means for storing.

17. The computer system as defined in claim 16 further comprising:

a first replicated program executing on the first means for executing, the first replicated program writing the read/write variable at a location indicated by the VMA in the page table of the first functional unit;

a second replicated program executing on the second means for executing, the second replicated program writing the read/write variable at a location indicated by the VMA in the page table of the second functional unit; and

wherein the first and second replicated programs are the copies of a same program.

18. The computer system as defined in claim 17 wherein the first and second replicated programs are copies of an operating system program, and wherein the read/write variable is a counter that indicates a number of executions of a code path of the operating system program.

19. The computer system as defined in claim 17 wherein the first and second replicated programs are copies of an operating system program, and wherein the read/write variable is a look-aside list header for process control blocks.

X. EVIDENCE APPENDIX

None.

XI. RELATED PROCEEDINGS APPENDIX

None.